Dreams of Lisbon

DECEMBER 2022

DATA MINING PROJECT REPORT

PROF. ROBERTO HENRIQUES

PROF. ANA AUBYN

PROF. LARA OLIVEIRA

GROUP 10

DIOGO ANTUNES M20221642
MARCO MORÁN M20220350
NELSON SOUSA R20191285
PEDRO SANTOS M20220069
RODRIGO FREIRE M20221292
VASCO MASSAPINA M20221294

Index

Methodology

CRISP-DM

To develop this data mining project, we are using the Cross-industry Standard Process for Data Mining (CRISP-DM). It's a powerful, practical, flexible methodology that helps to obtain effective results when using analytics to solve challenging business issues.

This method guided us through the process of the project, this also has the benefit of demonstrating a professional approach of the data mining project and relaying in a probed method to allow the projects to be replicated and understandable for everyone that read the document.

The cycle is composed of 6 steps:

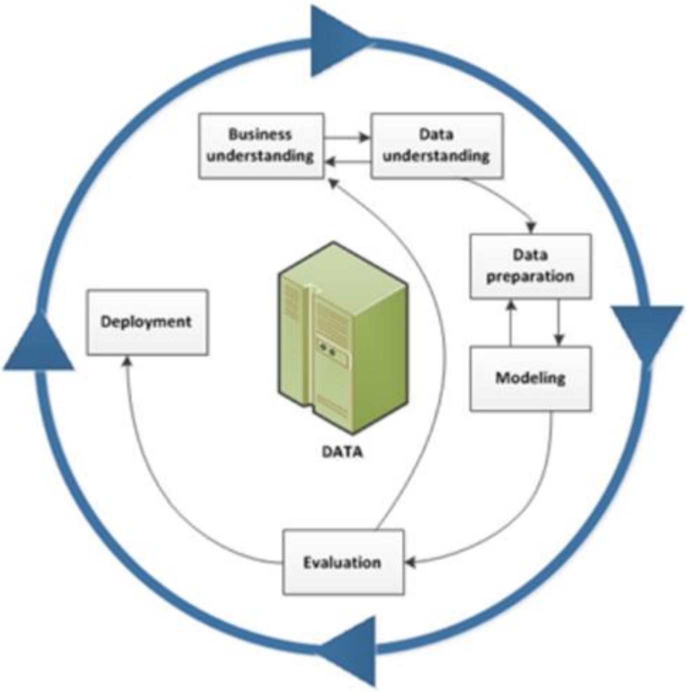


Figure 1 – CRISP-DM

Business Understanding

Now we will define the requirements and objectives of the project as well as the Data Mining problem, meaning that we will analyze what the company is doing and what the company needs to achieve, then we will set the goals of the project. In this phase, the goal is to uncover crucial factors that can influence the outcome of the whole project.

Data Understanding

This is the first approach to the data, we are able to check for correlations, identify missing values, understand the variables, and also accomplish a successful data importation and integration for the project purposes.

This is the phase where we can understand the quality of the data. We start by describing the Variables, which means examining the properties of the data that we are working with, then explore it and finally, we can verify the quality of the data.

Data Preparation

This is the phase where the analytical and operational work starts to meet because we are going to work on the conflicts of our Dataset. We start by selecting the data that we are going to use, and then the cleaning and transformation process need to be performed to have a dataset able to be modeled, this means constructing, integrating and formatting the data.

Modeling

In this stage, we select the best modeling technique by determining which algorithms we must use. Then we build the model that will help us reach the goals of the project.

Evaluation

This step is where we analyze if the information that we found after the modeling stage is reaching the business objectives. This phase is crucial because we will be able to know if we have to run new models or go back to previous phases of the methodology.

Deployment

When all the objectives of the project are met and the solutions to the data mining problems are found, we can deploy the result of the data mining modeling process through marketing plans, conclusions, or other options.

Business Understanding

The main objective is to develop and identify reasonable accommodation segmentation for a Portuguese company operating in the accommodation sector. “Dreams of Lisbon” works as an intermediary for multiple hotel chains and small companies, renting apartments, guesthouses, shared rooms, and hotel rooms.

To support direct marketing initiatives the company collected information and built a descriptive model by the data scientists.

Based on the segmentation of their accommodations this company can benefit from better marketing and sales decisions in terms of efficiency and cost reduction resulting in more satisfied clients.

Data understanding

Data and variables description

The first step to have a better understanding of the data that is provided "lisbonDreams.csv" is by checking the variables description. After removing a duplicate column of the variable "id" that had no name, the dataset is left with 19651 observations and 21 variables that are described in the following table.

Variable	Type	Description
id	int64	Identifier of the accommodation
start_date	object	Date from which the accommodation is in business
response_time	object	Indicator of how long it takes for the accommodation to answer clients online
response_rate	float64	Percentage of the messages received by the accommodation that get a reply
acceptance_rate	float64	Percentage of bookings made that the accommodation accepted
awards	object	Indicator of whether the accommodation was awarded for its quality
chain_hotels	int64	Number of accommodations managed by the same person/company
photos	object	Indicates if the accommodation has more than 20 photos available on the web page
verified	object	Indicates if the accommodation was already verified by Dreams of Lisbon
neighbourhood	object	Area where the accommodation stands
accommodation_type	object	Type of accommodation listed
accommodates	int64	Number of people that fit in the accommodation unit
bedrooms	float64	Number of bedrooms in the accommodation unit
price	float64	Price of a night in euros
number_of_reviews	int64	The number of reviews the listing has
first_review	object	Date of the first review
last_review	object	Date of the latest review
rating	float64	Average rating given by clients (0 to 5)
clean	float64	Average rating of cleanliness given by clients (0 to 5)
checkin	float64	Average rating of check-in process given by clients (0 to 5)
communication	float64	Average rating of ease of communication given by clients (o to 5)

Table 1 – Variables description

Descriptive statistics

The descriptive statistics approach must be done in two tables considering the type of our variables. We created a table for numerical variables and another for categorical variables that give us information and insights about the descriptive measures of each one considering the type of the attribute.

Variable	count	mean	std	min	25%	50%	75%	max	Missing Values
id	19651.000000	9825.000000	5672.899406	0.000000	4912.500000	9825.000000	14737.500000	19650.000000	0
response_rate	17835.000000	95.414466	14.042447	0.000000	99.000000	100.000000	100.000000	100.000000	1816
acceptance_rate	18296.000000	91.874399	19.234283	0.000000	95.000000	100.000000	100.000000	100.000000	1355
chain_hotels	19651.000000	50.729327	296.627535	0.000000	2.000000	6.000000	19.000000	6604.000000	0
accommodates	19651.000000	3.994199	2.407847	0.000000	2.000000	4.000000	5.000000	26.000000	0
bedrooms	18757.000000	1.752093	1.145923	1.000000	1.000000	1.000000	2.000000	22.000000	894
price	19651.000000	145.678641	304.477365	0.000000	70.380000	102.000000	155.040000	22440.000000	0
number_of_revies	19651.000000	54.666124	82.187743	0.000000	5.000000	21.000000	69.000000	1961.000000	0
rating	17717.000000	4.643287	0.457000	0.000000	4.530000	4.750000	4.900000	5.570000	1934
clean	17674.000000	4.692814	0.401319	0.000000	4.590000	4.800000	4.940000	5.000000	1977
checkin	17674.000000	4.778341	0.374709	-2.450000	4.740000	4.890000	4.980000	5.000000	1977
communication	17674.000000	4.785994	0.363621	1.000000	4.740000	4.900000	4.990000	5.000000	1977

Table 2 – Descriptive statistics for numerical variables

According to the descriptive statistical analysis above we noticed that there are several steps to take to have the data in a good state to work with. Due to the difference between the minimum and the percentil 25 and the difference between the maximum value and the percentil 75 we can see that a few variables have outliers. In the case of the "chain_hotels" the percentil 75 is 19 while the maximum value is 6604 and the standard deviation (296.627535) in comparison to the mean (50.729327) in this variable is also a sign of outliers. The "accomodates" percentil 75 is 5 while the maximum value is 26. The "bedrooms" percentil 75 is 2 and the maximum 22. The variable "price" has 155.04 on percentil 75 and 22440 on maximum value. The "number_of_reviews" percentil 75 is 69 and the maximum 1961. With this analysis we can say that these 5 variables have a high probability of having upper outliers.

One of the things that does not make sense is the minimum value of the variable "price" being 0€, this means an incoherence for further checking. There seems to be also an incoherence with the "rating" and "checkin" values, our index tells us that these values can only go from 0 to 5 but the minimum of variable "checkin" is -2.45 and the maximum value of "rating" is 5.57.

Another thing that deserves closer attention are the significant amount of missing values on the variables "response_rate", "acceptance_rate", "bedrooms", "rating", "clean", "checkin" and "communication".

Variables	Count	Uniq	Top	Freq	Missing Values
start_date	19651	3288	2012-10-23	299	0
response_time	17835	4	within an hour	14373	1816
awards	19644	3	No	14054	7
photos	19651	2	Yes	19529	0
verified	19651	2	Yes	17007	0
neighbourhood	19651	6	Lisbon	13629	0
accommodation_type	19651	4	Appartment	15096	0
first_review	17717	3139	2019-04-21	54	1934
last_review	17717	1244	2022-08-29	624	1934

Table 3 – descriptive statistics for categorical variables

Table x - Descriptive Statistics for Categorical Variables

In terms of categorical variables the missing values deserve our attention. As we can check the count number standard should be 19651 and so the “response_time”, “awards”, “first_review” and “last_review” are the variables with missing values.

Coherence checking

To make sure the data we are studying doesn’t suffer incoherences that could lead to inferior performance for our modeling techniques, we decided to analyze the data for some coherence checks. Firstly, we decided to make sure there were no duplicate observations. Moreover, after analyzing the dataset we spotted what seems to be an incoherence regarding the variables “rating” and “check-in”*.* These two variables are results from a survey conducted among the customers, with values that range from 0 to 5 according to the metadata that was given. Therefore, we noticed on the variable *rating* a max value of 5.57 and a min value of -2.45 on the “check-in” variable, which doesn’t make sense, and to rectify this anomaly we decided to restrain the range by replacing every value higher than 5 with 5 and every negative value will be replaced with 0.Secondly, we noticed two bedrooms were priced at 0€, which doesn’t make sense as well and therefore we decided to remove them since it's only two observations, it's not going to make a lot of difference on our study.

Detect Outliers

When an observation lies an abnormal distance from other values in a random sample from a population, it's called an outlier.

As we explained in Data Preparation, we identified 'chain_hotels', 'accommodates', 'bedrooms', 'price' and 'number_of_reviews' were the variables that had the potential presence of outliers.

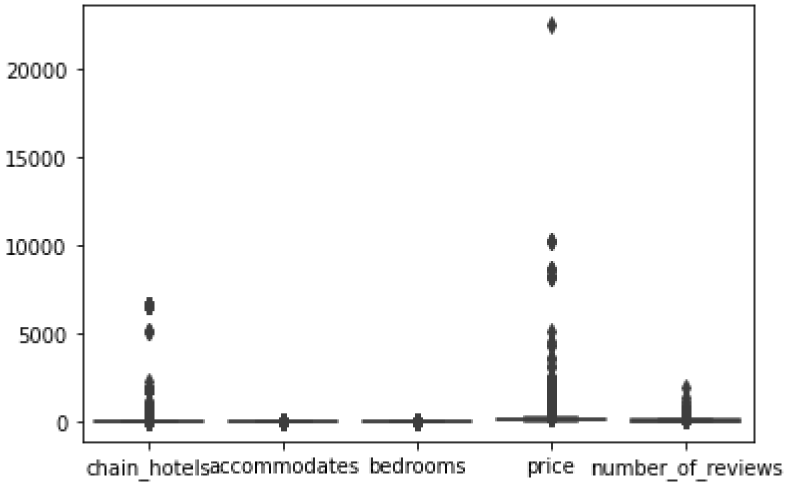


Figure 2 – Boxplot with potential outliers

Looking at the boxplot, we can see that 'chain_hotels' and 'price' have some outliers that diverge a lot from the remaining entries

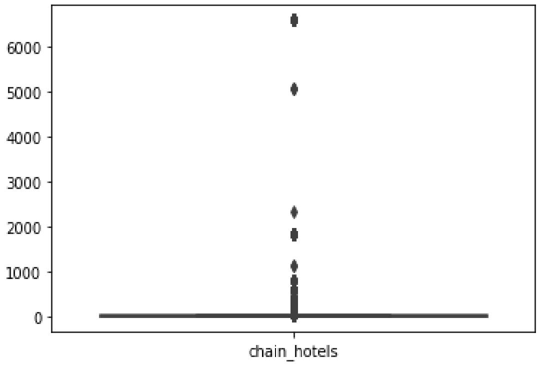


Figure 3 – 'Chain Hotels' boxplot

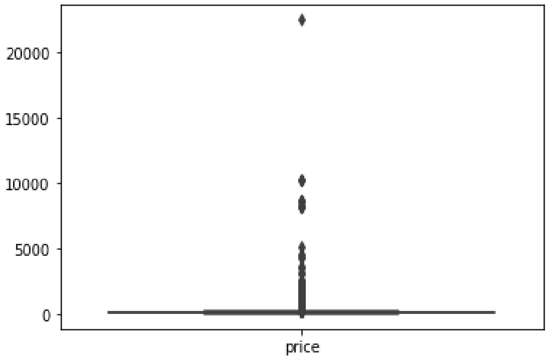


Figure 4 – 'price' boxplot

Data preparation

Outliers Treatment

We created a new dataset named **lb4** that will be equal to the **lb3** dataset, but only for the rows where ‘chain_hotels’ is not considered an outlier according to the boxplot, so for that, we decreased the variable (´chain_hotels´) length to a max of 1300.

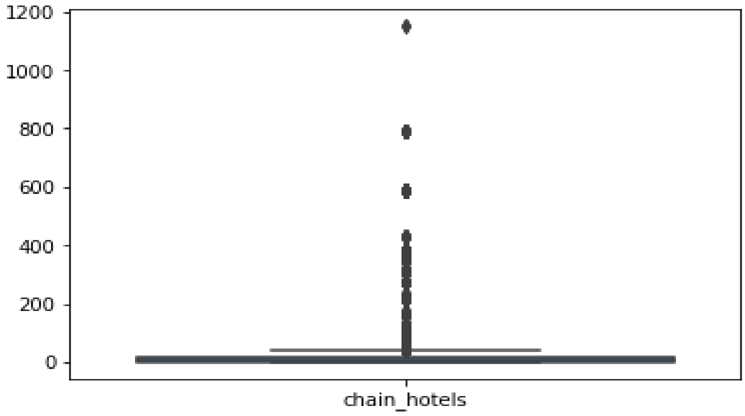


Figure 5 – ‘Chain hotels’ boxplot

Now for the variable ´price´, we decided to decrease the length to a max of 6000, and decided to drop the other values that were above that limit because we identified them as outliers.

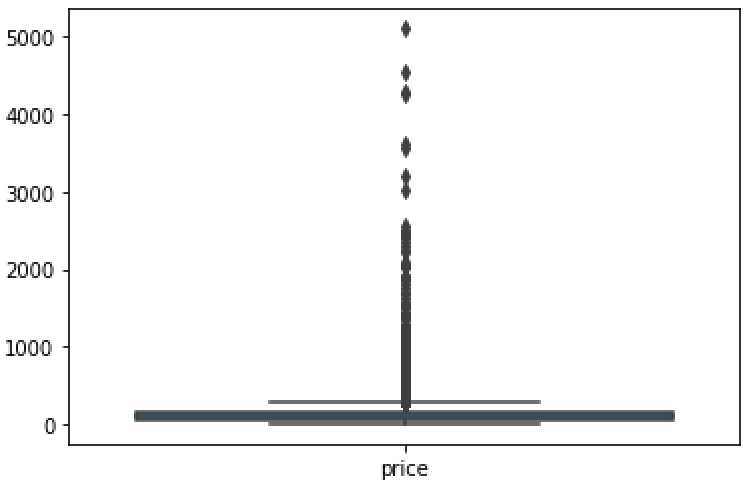
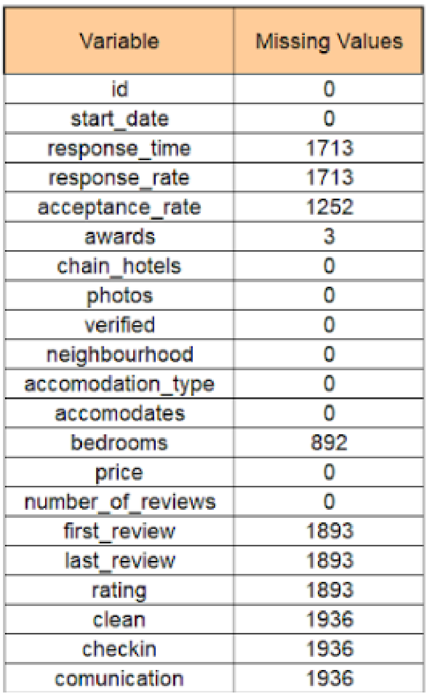


Figure 6 - ´Price’ boxplot

Checking for Missing Values

As we start analyzing the data, we start to check if there were any missing values in the dataset. To do this, we can use the methods *isna()* and *sum()* combined as a way to obtain a list of the variables containing missing values.



Variable	Missing Values
id	0
start_date	0
response_time	1713
response_rate	1713
acceptance_rate	1252
awards	3
chain_hotels	0
photos	0
verified	0
neighbourhood	0
accomodation_type	0
accomodates	0
bedrooms	892
price	0
number_of_reviews	0
first_review	1893
last_review	1893
rating	1893
clean	1936
checkin	1936
comunication	1936

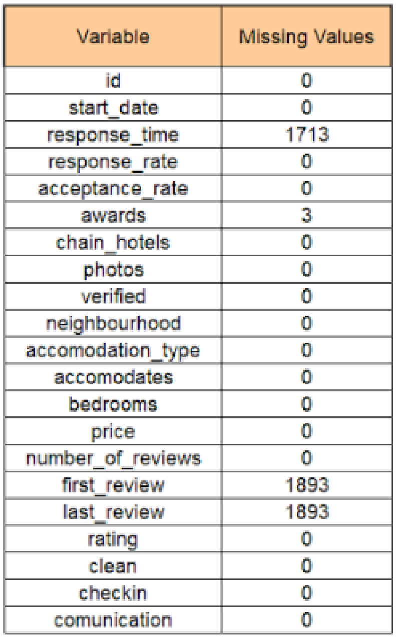
Table 4 – Identify Missing values

After analyzing the results, we can conclude we have missing values on the columns: 'response_time', 'response_rate', 'acceptance_rate', 'awards', 'bedrooms', 'first_review', 'last_review', 'rating', 'clean', 'checkin', 'communication'.

Missing Values Treatment

One way of dealing with rows that have missing values is to drop the rows, using the method *dropna()*, but in this case, we decided to take a different approach.

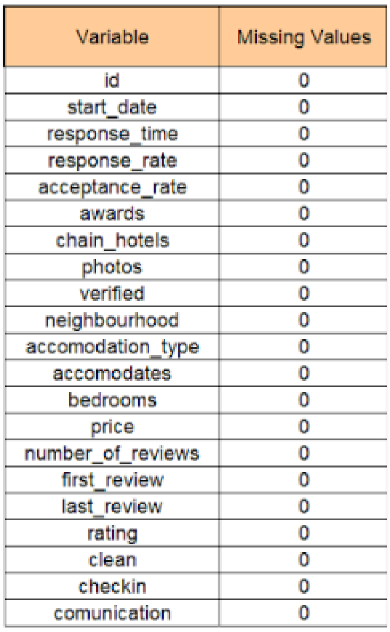
In order to treat those missing values, we decided to fill in those missing values using the method ***fillna()*** and we did that for the accommodation types that are hotel room and shared room, because all of them can be just one room, so in the case that the number of bedrooms are missing it will replace them with the value 1 and for the other variables, we define the parameter **value = lb6.median()** and created a new dataset named **lb6.fillna** when applying this replacement.



Variable	Missing Values
id	0
start_date	0
response_time	1713
response_rate	0
acceptance_rate	0
awards	3
chain_hotels	0
photos	0
verified	0
neighbourhood	0
accomodation_type	0
accomodates	0
bedrooms	0
price	0
number_of_reviews	0
first_review	1893
last_review	1893
rating	0
clean	0
checkin	0
comunication	0

Table 5 – Missing value treatment

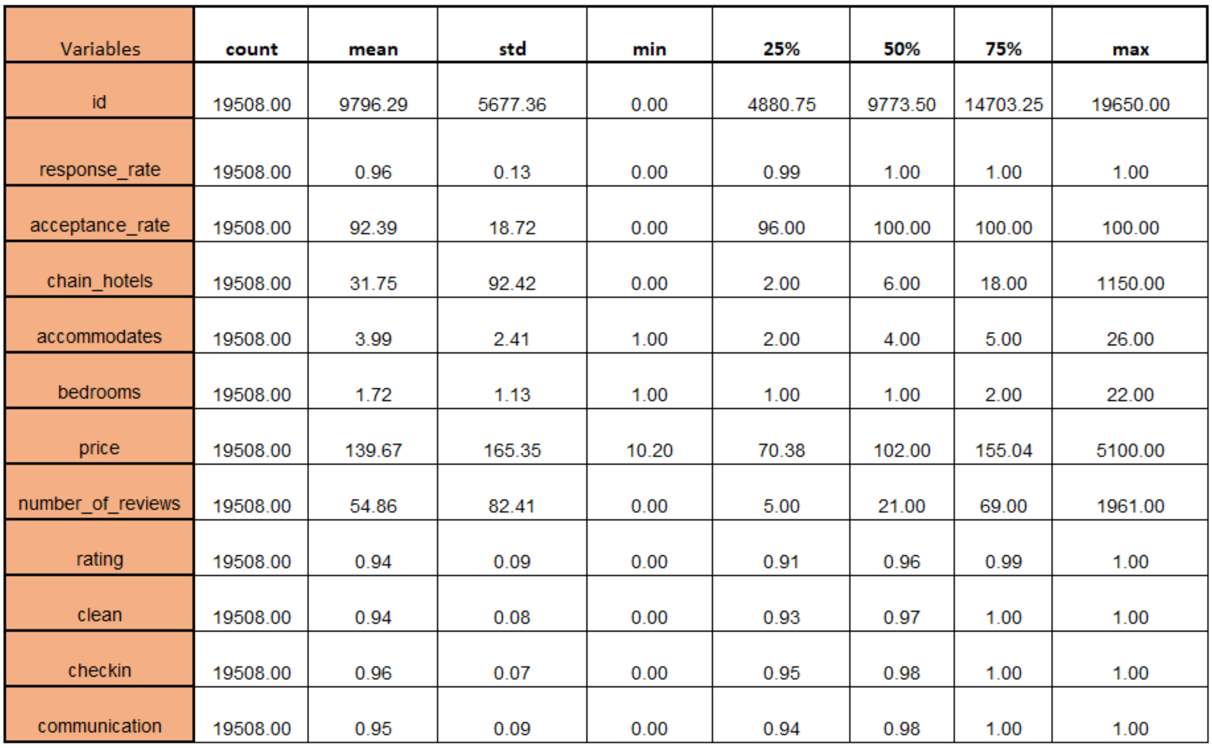
An easier way to fill the categorical variables is making python determine mode by itself.



Variable	Missing Values
id	0
start_date	0
response_time	0
response_rate	0
acceptance_rate	0
awards	0
chain_hotels	0
photos	0
verified	0
neighbourhood	0
accomodation_type	0
accomodates	0
bedrooms	0
price	0
number_of_reviews	0
first_review	0
last_review	0
rating	0
clean	0
checkin	0
comunication	0

table 6 – Missing value treatment

Descriptive Statistics After Data Treatment



Variables	count	mean	std	min	25%	50%	75%	max
id	19508.00	9796.29	5677.36	0.00	4880.75	9773.50	14703.25	19650.00
response_rate	19508.00	0.96	0.13	0.00	0.99	1.00	1.00	1.00
acceptance_rate	19508.00	92.39	18.72	0.00	96.00	100.00	100.00	100.00
chain_hotels	19508.00	31.75	92.42	0.00	2.00	6.00	18.00	1150.00
accommodates	19508.00	3.99	2.41	1.00	2.00	4.00	5.00	26.00
bedrooms	19508.00	1.72	1.13	1.00	1.00	1.00	2.00	22.00
price	19508.00	139.67	165.35	10.20	70.38	102.00	155.04	5100.00
number_of_reviews	19508.00	54.86	82.41	0.00	5.00	21.00	69.00	1961.00
rating	19508.00	0.94	0.09	0.00	0.91	0.96	0.99	1.00
clean	19508.00	0.94	0.08	0.00	0.93	0.97	1.00	1.00
checkin	19508.00	0.96	0.07	0.00	0.95	0.98	1.00	1.00
communication	19508.00	0.95	0.09	0.00	0.94	0.98	1.00	1.00

table 7 – Descriptive statistics after data treatment

After the treatment of the missing values and the outliers that were present in our data set, we can see that the count for our variables is now all the same at 19508 due to the treatment and input of the missing values.

We can also check on the 75% percentile and the Max values from the 2 variables, ‘chain_hotel’ and ‘price’, are now, respectively 18 (Percentile 75), 1150 (Max) and 155.04 (Percentile 75), 5100 (Max), due to the cleaning of outliers in those variables while reducing their length.

Correlations

To analyze the correlations between the variables after the data treatment, we used the Heatmap to process the information in a visual way, going from the ones with the most correlation (-1 and 1) with a lighter color in value 1 and a darker color when going to -1.

We set up the limits going from -1 to 1, meaning that when the correlation between two variables goes closer to 1 (having a positive correlation), as one variable increases so does the other one. When a variable gets closer to -1, being a negative correlation, as one variable increases, the other decreases and when the value gets closer to 0, their relationship becomes increasingly nonexistent, meaning that the movement of a variable cannot be predicted from the other.

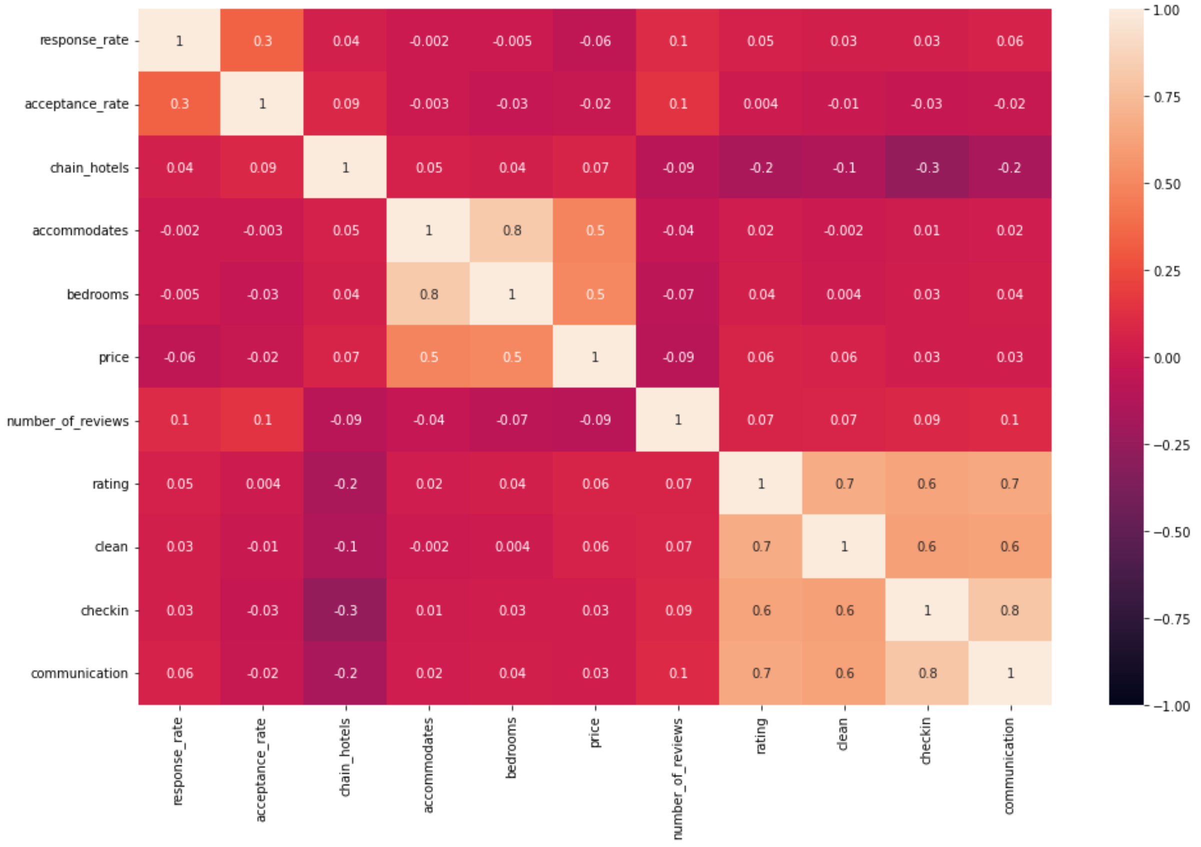


Figure 7 – correlation heatmap

In this figure, we can take conclusions for the relationships between different variables, such as ‘Communication’ and ‘Checkin’; ‘Bedrooms’ and ‘Accommodates’; with the highest positive correlation (0.8), meaning we can assume that as one variable increases, so does the other one. The lowest negative correlation (-0.3) is between the variables ‘CheckIn’ and ‘Chain_Hotels’.

Modelling

Clustering

Clustering refers to the grouping of records, observations on cases into classes of similar objects. This means, from a set of data, organize them in homogenous groups, determining a “structure” of similarities between units.

K-Means Clustering

K-means is an unsupervised machine learning algorithm used for clustering. It’s a method of dividing a dataset into a predetermined number of clusters (also called "k") in a way that minimizes the sum of the distances between each data point and the mean (centroid) of its assigned cluster.

One of the main advantages of k-means is its simplicity and speed, which makes it suitable for large datasets.

When applying the k-means, we first had an overview of the data we were working with after its pre-processing. Following that, we defined what variables we were going to work with, since the objective was to evaluate customer satisfaction, and we ended up choosing “rating”, for how good the room was, “clean”, for how clean the room was, “check-in”, for how easy the check-in was, and “communication”, for how easy it was to communicate with the staff.

Continuing our task, we then used a library called “sklearn.cluster” to import the “K-means” algorithm and proceed to utilize the “elbow method” and the “Dendrogram to try and identify the right number of clusters. We started with the “elbow method”, where we try to identify the right number of clusters based on the results of the K-means inertia graph. To use the elbow method, we must calculate the sum of squared errors (SSE) between each data point and the centroid of the cluster to which it was assigned.

Then, we plot the values of k against the respective obtained SSE. As k increases, the SSE decreases.

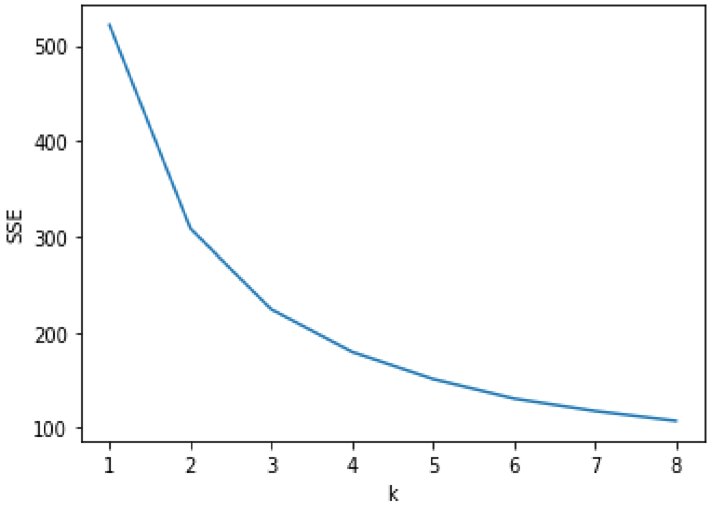


Figure 8 – Elbow method

We couldn't find any conclusive information, so we tried a Dendrogram.

In the context of k-means, a dendrogram can be used to visualize the relationships between different clusters and to help determine the optimal number of clusters (k) for the data.

By examining the dendrogram, you can see how the data points are grouped into clusters and how the clusters are related to each other. You can also use the dendrogram to identify the optimal number of clusters for k-means by looking for the maximum distance vertically on the hierarchical cluster without intersecting another cluster.

After constructing the hierarchical clustering dendrogram, we got the following graph:

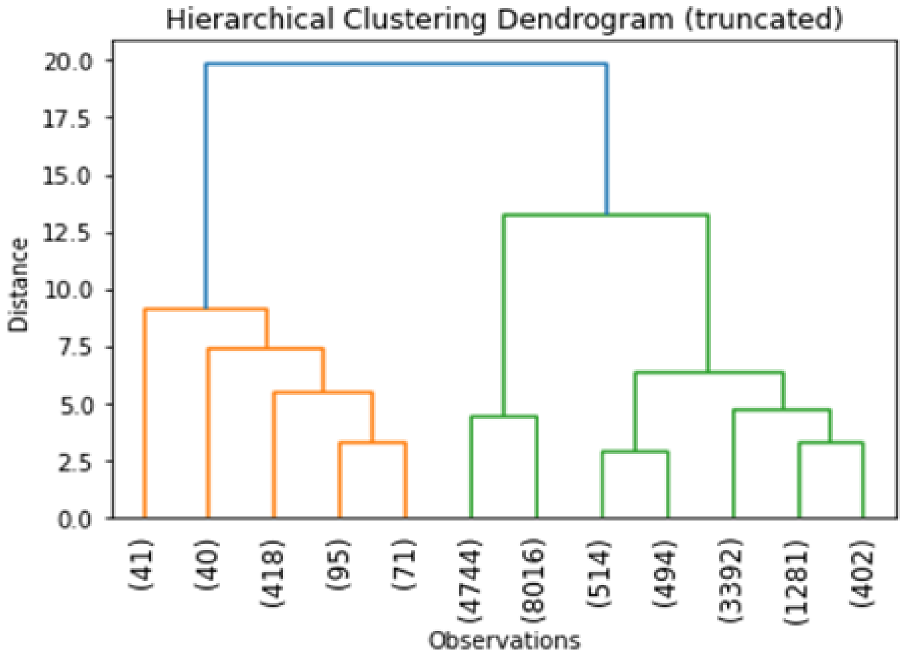


Figure 9 – Hierarchical clustering Dendogram

So, after analyzing both graphs, looking to the Dendrogram we can see the higher vertical distance is from around 9 to 14, so drawing two horizontal lines we can see 3 vertical lines crossing, so it means that the best number of clusters is 3.

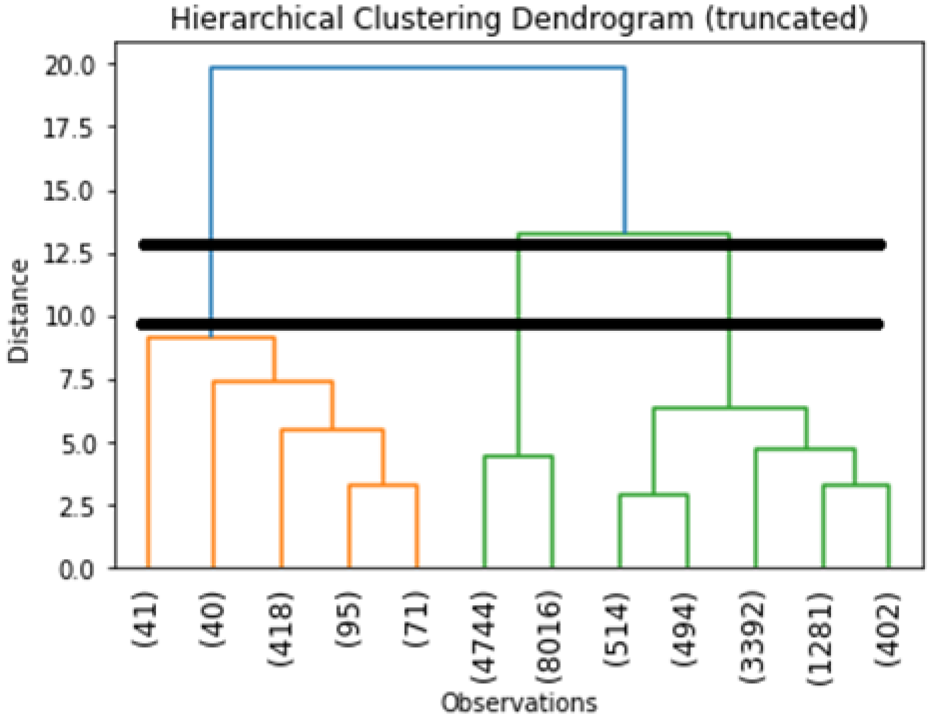
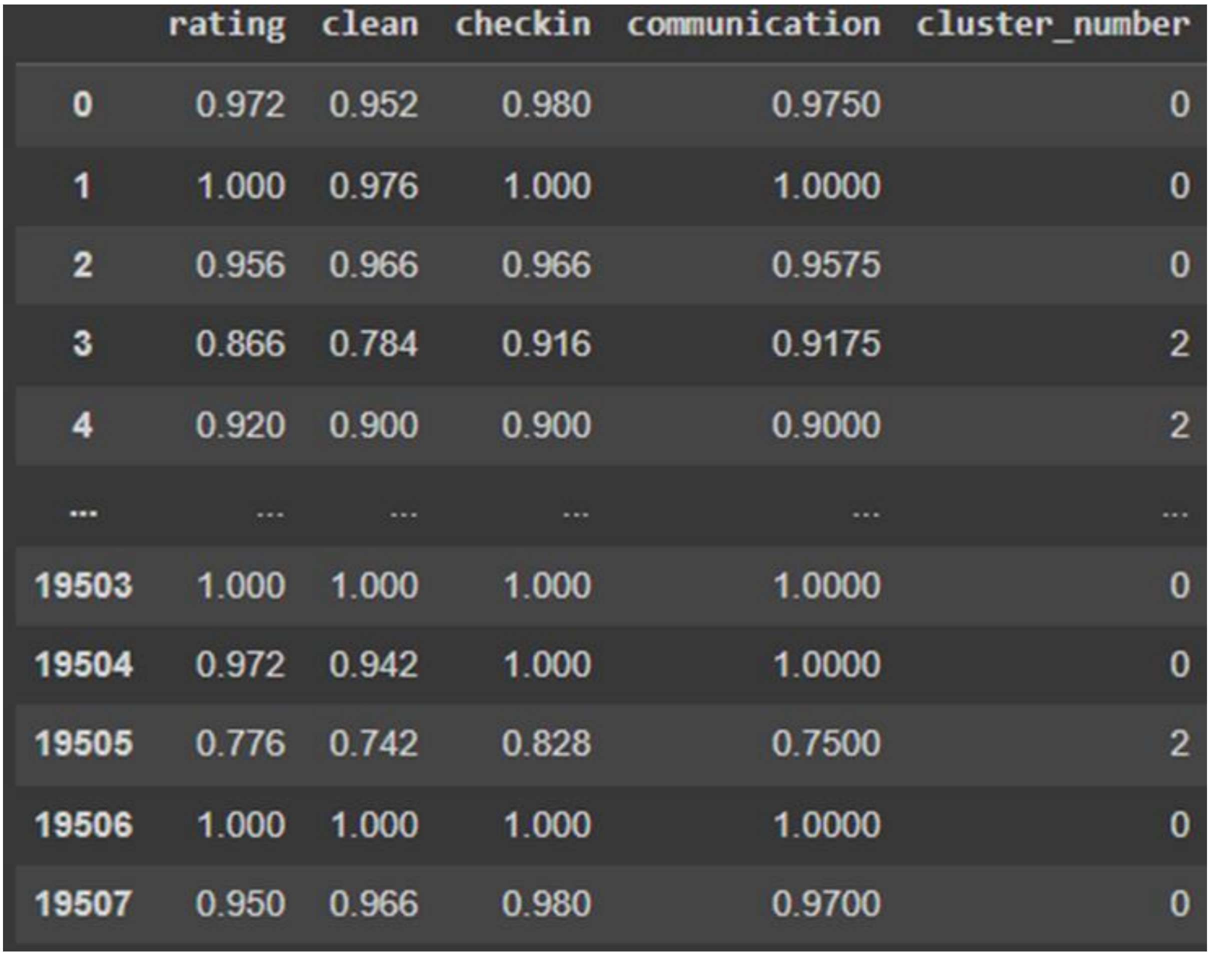


Figure 10 - Hierarchical clustering Dendogram

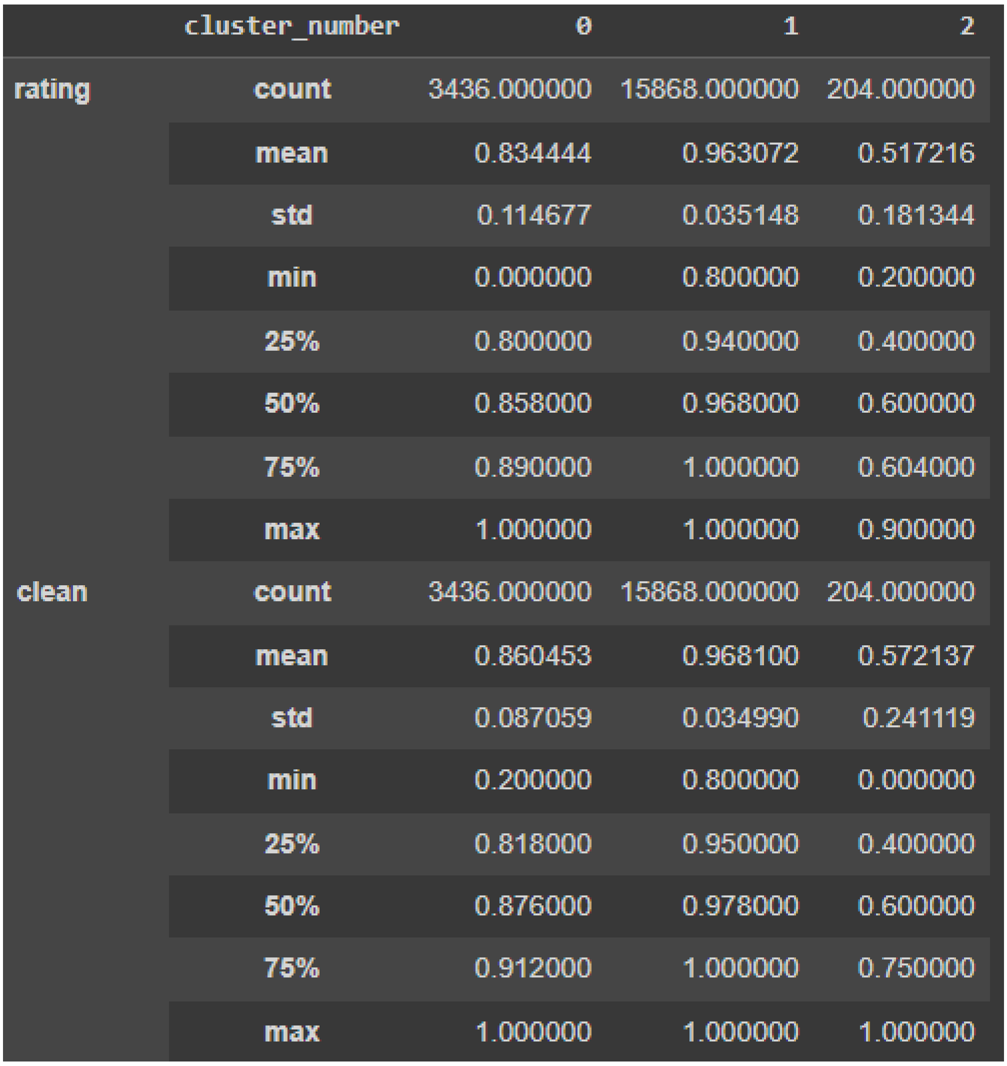
Now that we know the number of cluster ideal for our data set, we proceed with the K-means algorithm. We made an extra variable named “cluster_number” to better interpret the results.



	rating	clean	checkin	communication	cluster_number
0	0.972	0.952	0.980	0.9750	0
1	1.000	0.976	1.000	1.0000	0
2	0.956	0.966	0.966	0.9575	0
3	0.866	0.784	0.916	0.9175	2
4	0.920	0.900	0.900	0.9000	2
...	...	...	...	...	...
19503	1.000	1.000	1.000	1.0000	0
19504	0.972	0.942	1.000	1.0000	0
19505	0.776	0.742	0.828	0.7500	2
19506	1.000	1.000	1.000	1.0000	0
19507	0.950	0.966	0.980	0.9700	0

The “cluster_number” variable varies between 0, 1 and 2 because that’s the number of clusters we have.

We then checked the descriptive statistics for each of the clusters and concluded that cluster 0 is much more concentrated than cluster 1 and 2, however the STD explains why this could be happening. On cluster 0 the STD is exceedingly small which means that the values are remarkably close to each other and that explains the large amount of data of that cluster.



	cluster_number	0	1	2
rating	count	3436.000000	15868.000000	204.000000
	mean	0.834444	0.963072	0.517216
	std	0.114677	0.035148	0.181344
	min	0.000000	0.800000	0.200000
	25%	0.800000	0.940000	0.400000
	50%	0.858000	0.968000	0.600000
	75%	0.890000	1.000000	0.604000
	max	1.000000	1.000000	0.900000
clean	count	3436.000000	15868.000000	204.000000
	mean	0.860453	0.968100	0.572137
	std	0.087059	0.034990	0.241119
	min	0.200000	0.800000	0.000000
	25%	0.818000	0.950000	0.400000
	50%	0.876000	0.978000	0.600000
	75%	0.912000	1.000000	0.750000
	max	1.000000	1.000000	1.000000

We then wanted to visualize the cluster after performing the k-means, so, for that, we did a distribution of the plot for the three clusters, and then we plotted a graph for the variable “ratings” and one for the variable “clean”.

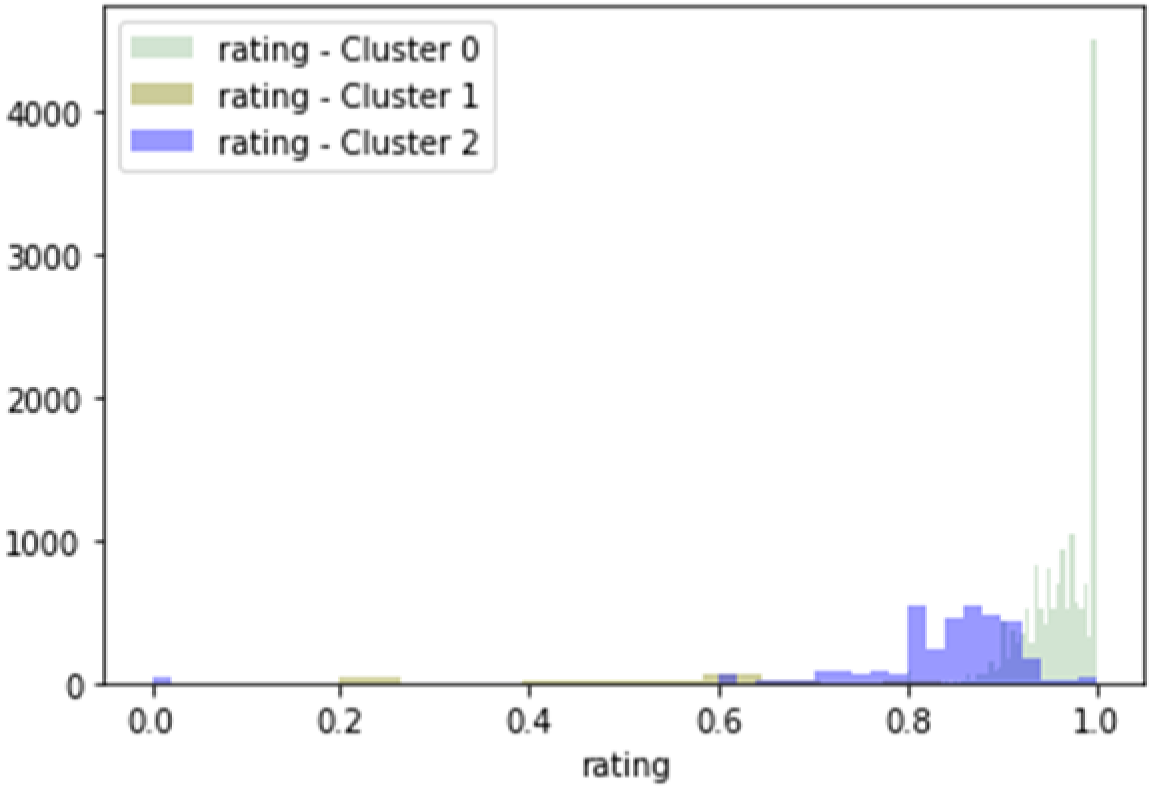


Figure 11 – cluster visualization for 'rating' variable

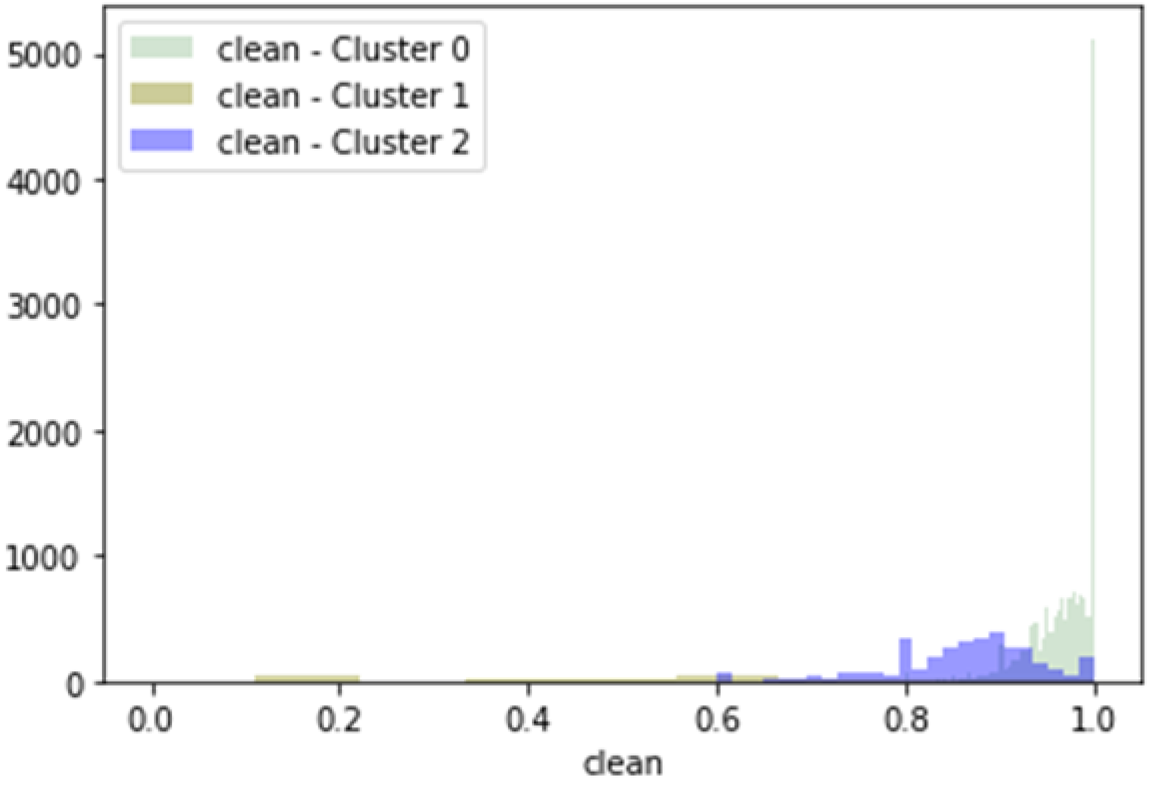


Figure 12 – Cluster visualization for 'clean' variable

And then, we made a scatter plot for the 3 clusters, where we compare the "rating" and the "clean" variables.

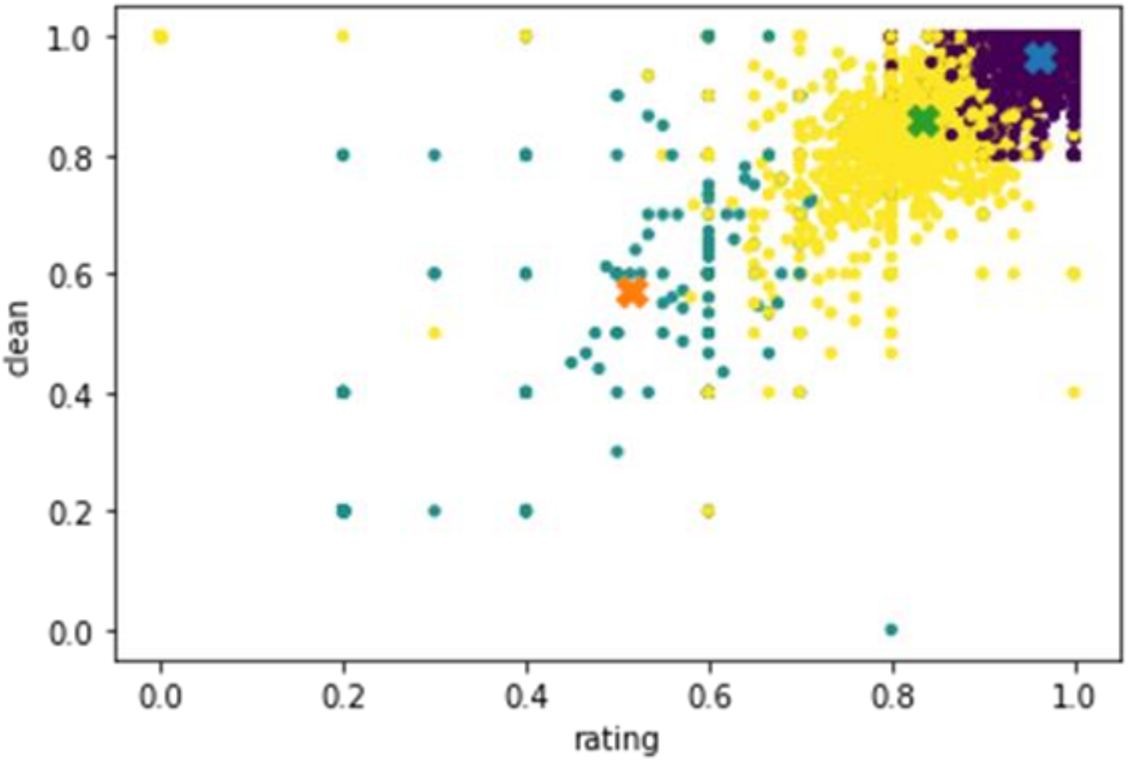


Figure 13 – scatter plot

We can see that the STD on cluster 0 is extremely low, which means that all the observations have similar values, explaining why cluster 0 is so dense.

From the results that we studied, we believe that accommodations present on cluster 1 should be improved and upgraded because the mean of the "rating" is incredibly low compared to cluster 0 and 2. Regarding cluster 0 and 2, we should focus on promoting the accommodation present in both clusters because they're our best accommodations and are more reliable to generate better results in our marketing campaigns.

SOM

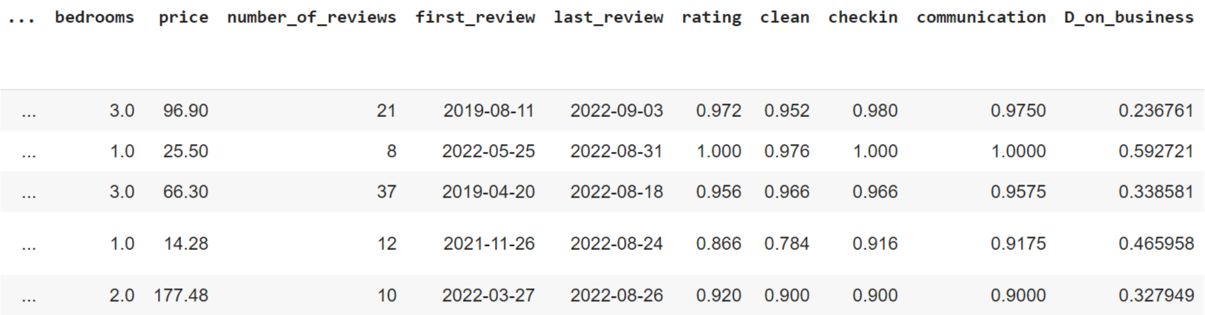
Self Organizing Maps (SOM) is an algorithm based on neural networks that is used to represent a multi-dimensional dataset into a two-dimensional pattern, meaning that all the complex non-linear relationships between the observations of different variables become simple geometric relationships that can be displayed in two dimensions.

Next we are going to show the step by step developed by the group of the SOM algorithm using the “Lisbon Dreams” dataset, in this case we are seeking to create a clustering segmentation of the accommodations in the data, based on a specific perspective of the data.

As a result of the data preparation step, we obtained a clean dataset called “LisbonDreams_preproF.csv” which is the data that we will be working with in this algorithm. It’s important to state that the data within the dataset is scaled from 0 to 1.

After a first look into the dataset we found a possible correlation between the variables and a specific column of data, which is the date that the accommodation started to offer the service, In order to know how many days the accommodation has been in the market, we need to transform the variable "start_date" to a numerical date, also creating a new column with the number of days with this concept. then we must scale these values so we can analyze it with the other variables

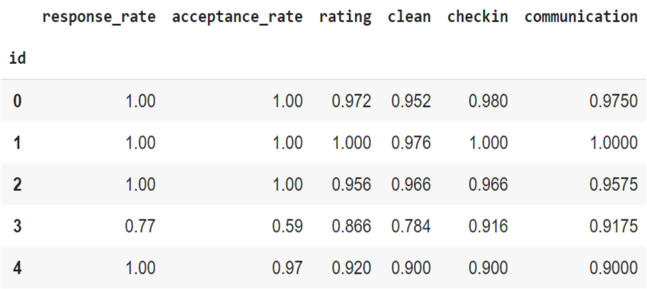
The resulting data frame is as follows:



...	bedrooms	price	number_of_reviews	first_review	last_review	rating	clean	checkin	communication	D_on_business
...	3.0	96.90	21	2019-08-11	2022-09-03	0.972	0.952	0.980	0.9750	0.236761
...	1.0	25.50	8	2022-05-25	2022-08-31	1.000	0.976	1.000	1.0000	0.592721
...	3.0	66.30	37	2019-04-20	2022-08-18	0.956	0.966	0.966	0.9575	0.338581
...	1.0	14.28	12	2021-11-26	2022-08-24	0.866	0.784	0.916	0.9175	0.465958
...	2.0	177.48	10	2022-03-27	2022-08-26	0.920	0.900	0.900	0.9000	0.327949

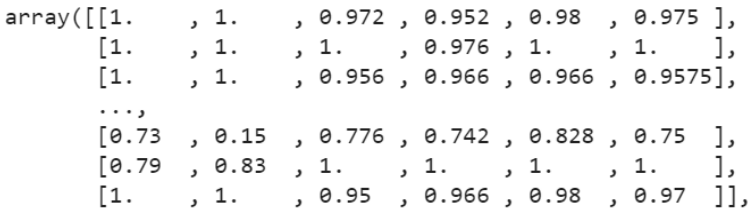
The use of this new variable called “D_on_business” is for analysis purposes, meaning that this variable will not be used in the performance of the algorithm, but will be used in the analysis of the clusters. The decision of not using this variable in the SOM algorithm is because even though these values are scaled, the clustering seems really affected by these observations.

The next step is selecting a perspective. For this algorithm we chose the variables: 'response_rate','acceptance_rate','rating','clean','check-in','communication'.This perspective was chosen because they contextualize the client satisfaction. The data frame that we created is called “df_persp”.



id	response_rate	acceptance_rate	rating	clean	checkin	communication
0	1.00	1.00	0.972	0.952	0.980	0.9750
1	1.00	1.00	1.000	0.976	1.000	1.0000
2	1.00	1.00	0.956	0.966	0.966	0.9575
3	0.77	0.59	0.866	0.784	0.916	0.9175
4	1.00	0.97	0.920	0.900	0.900	0.9000

After deciding about the perspective that we are going to pursue with this algorithm, we are going to create the Self Organizing Maps instance.



```
array([[1.    , 1.    , 0.972 , 0.952 , 0.98  , 0.975 ],
       [1.    , 1.    , 1.    , 0.976 , 1.    , 1.    ],
       [1.    , 1.    , 0.956 , 0.966 , 0.966 , 0.9575],
       ...,
       [0.73  , 0.15  , 0.776 , 0.742 , 0.828 , 0.75  ],
       [0.79  , 0.83  , 1.    , 1.    , 1.    , 1.    ],
       [1.    , 1.    , 0.95  , 0.966 , 0.98  , 0.97  ]]),
```

The SOM algorithm has different parameters that need to be set, first we need to define the size of the grid, then we set how SOM will behave. For example, we must choose the “mapshape” parameter which is the shape of the grid. We tried different options of the parameters to make the results accurate and easier to understand.

The next step is the training part of the algorithm where we define the number of times that the algorithm will run the process.

Now we must create an Umatrix to look at the grid with all the variables that we chose. This plot is important because based on this we can determine how many clusters we will be using for the k-means algorithm that we will apply next to our dataset.

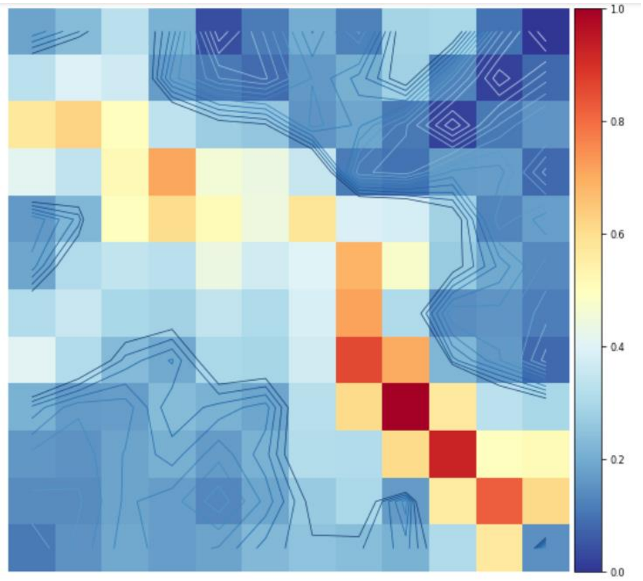


Figure 14 - Umatrix

To have a better understanding of the result, we create component planes of the som, meaning that through heatmaps we are going to be able to see the result for each dimension or variable in our perspective.

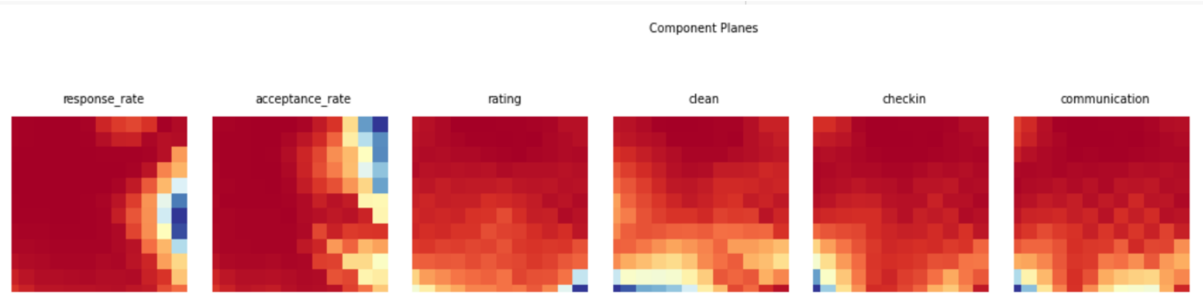
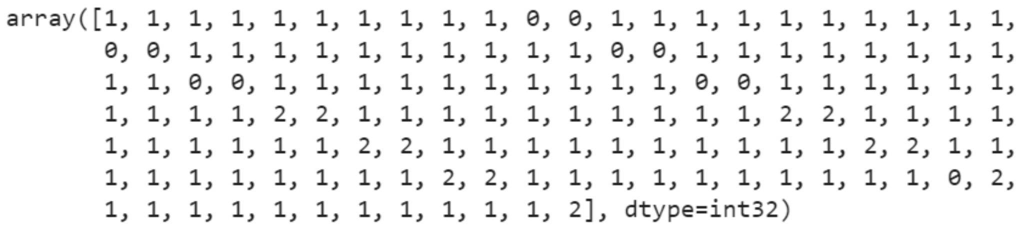


Figure 15 – Heatmap visualization of each variable

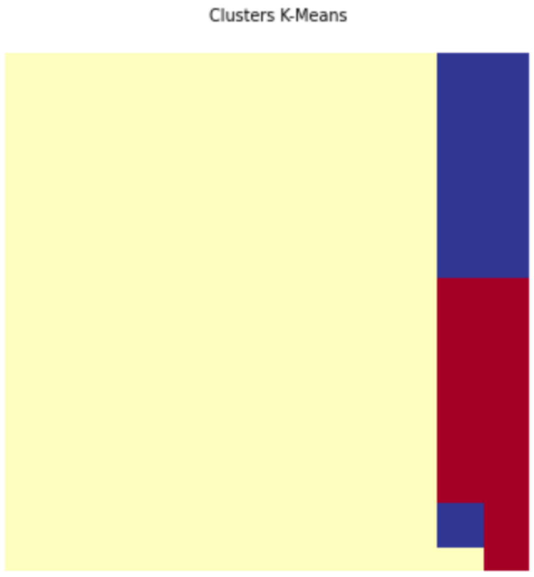
After finding insights in the visual part of the algorithm, we call the best matching units (BMU) of each observation, and also the codebook.matrix to get the values of the weight that each neuron has, is important to gather this information in order to be able to label our dataset.

The next step is to apply the K-means algorithm, with all the information that we acquired during the SOM, we can decide how many clusters we should have in the algorithm. Also, we are getting the labels of each cluster associated by K-means to each neuron in the SOM map.

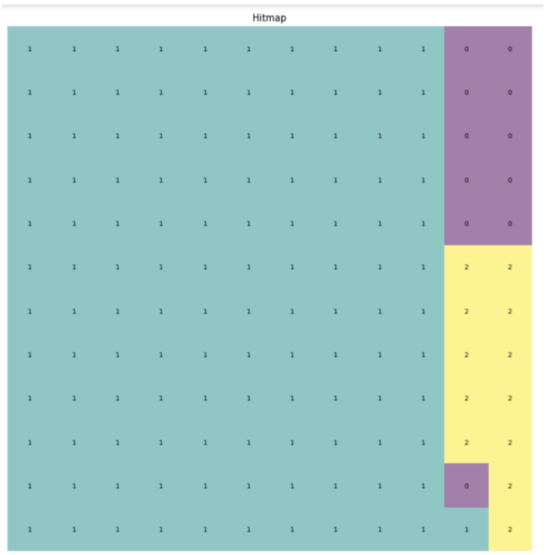


```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 2,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2], dtype=int32)
```

Then we created plots for the results of the K-means algorithm with our 3 clusters.

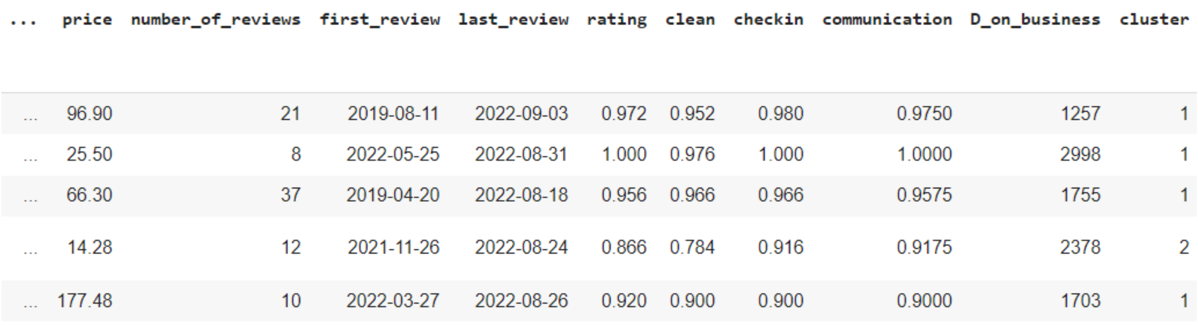


Clusters K-Means

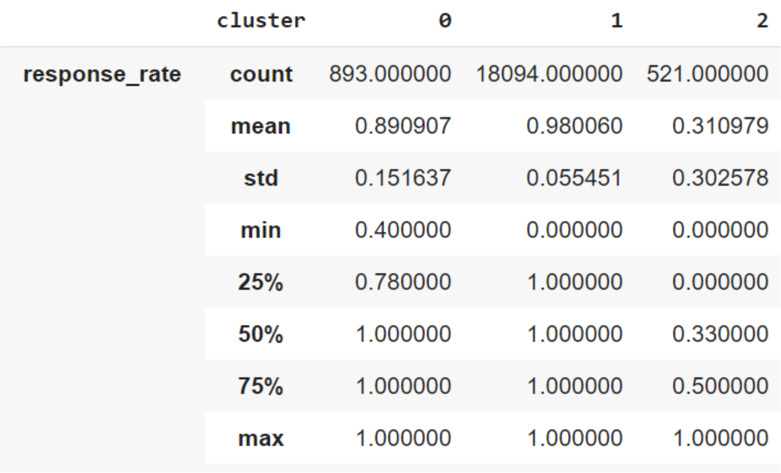


Hitmap

The Final step is adding the label of the clusters for each accommodation in the original data frame and obtaining the descriptive statistics for clusters in each variable to analyze the data, look for correlations and also find insights to make proposals for the business.



...	price	number_of_reviews	first_review	last_review	rating	clean	checkin	communication	D_on_business	cluster
...	96.90	21	2019-08-11	2022-09-03	0.972	0.952	0.980	0.9750	1257	1
...	25.50	8	2022-05-25	2022-08-31	1.000	0.976	1.000	1.0000	2998	1
...	66.30	37	2019-04-20	2022-08-18	0.956	0.966	0.966	0.9575	1755	1
...	14.28	12	2021-11-26	2022-08-24	0.866	0.784	0.916	0.9175	2378	2
...	177.48	10	2022-03-27	2022-08-26	0.920	0.900	0.900	0.9000	1703	1



	cluster	0	1	2
response_rate	count	893.000000	18094.000000	521.000000
	mean	0.890907	0.980060	0.310979
	std	0.151637	0.055451	0.302578
	min	0.400000	0.000000	0.000000
	25%	0.780000	1.000000	0.000000
	50%	1.000000	1.000000	0.330000
	75%	1.000000	1.000000	0.500000
	max	1.000000	1.000000	1.000000

DBSCAN

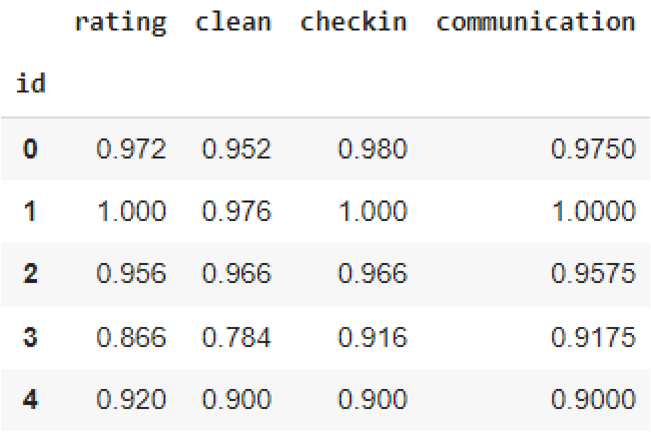
DBSCAN algorithm stands for Density-based Spatial Clustering of Applications with noise and is a well-known algorithm that is very used in data mining. It uses the data distribution to find and define areas with high density and uses those areas to define the boundaries of each cluster, DBSCAN groups points that are close to each other based on distance and a minimum number of points, which are the two parameters:

- **Epsilon:** how close points should be
- **minPoints:** minimum number of points to form a dense region

Using the “Lisbon Dreams” dataset, the group is going to implement the DBSCAN to create a clustering segmentation of the accommodation based on the perspective defined before.

Using the already preprocessed and scaled dataset “LisbonDreams_preproF.csv”, first we needed to define the index, so we used the set_index method to set the “id” column as an index.

After defining “id” as index, we selected the variables that are relevant for our analysis, considering the project perspective, which are: rating, clean, check-in and communication. We called it “df_prod”:



id	rating	clean	checkin	communication
0	0.972	0.952	0.980	0.9750
1	1.000	0.976	1.000	1.0000
2	0.956	0.966	0.966	0.9575
3	0.866	0.784	0.916	0.9175
4	0.920	0.900	0.900	0.9000

After preparing the data that we are going to use we imported the needed libraries, sklearn.cluster and sklearn.preprocessing, and everything is ready to run the algorithm.

We should then define the parameters, so in the beginning we defined the eps = 0.15 and the minSamples = 5, then we used the describe method to analyze our data, create the visualization for the variables rating and cleaning:

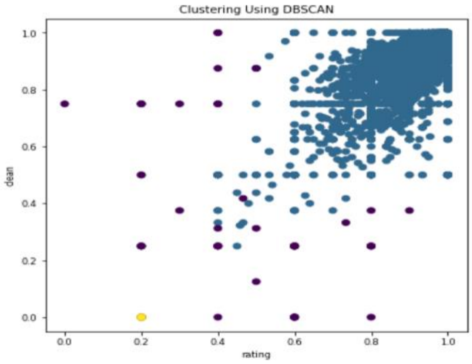


Figure 16 – visualization for the variables 'rating' and 'cleaning'

and on the next step, we created visualizations to compare all variables with each other:

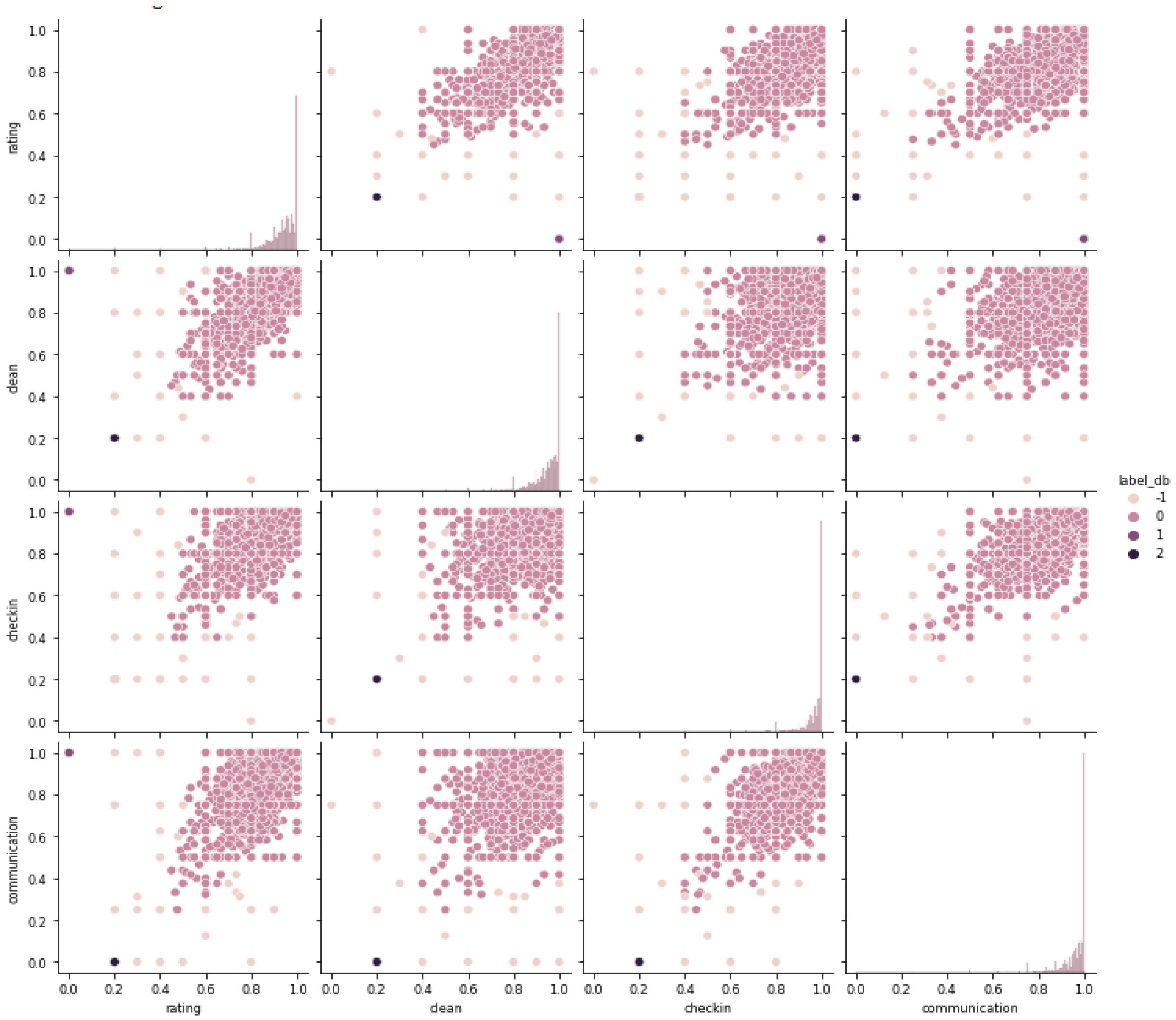


Figure 17 – visualization for all the variables

After the first try we changed the parameters to different values, and we saw that the number of clusters would increase too much, and each cluster would have a small count and the cluster 0 kept the same count number. This happens because the data is so concentrated in one place and so dense that only two scenarios would happen:

1. If we increase the parameters, for example the eps, we would get only one cluster and the outliers. The reason behind is that with a bigger radius, DBSCAN has a higher range which means that will capture more points and

since the data is so dense and concentrated the algorithm would define all as a big cluster

2. If we decreased the parameters, we would get a substantial number of clusters because the process would invert, this means that the radius now is small and with less minSamples that the algorithm would cluster almost every point as a cluster.

After this analysis, we decided to keep the eps = 0.15 and minSamples = 5 since we felt that would be the best match and the best way to analyze and get relevant data.

Analyzing the final output, we can conclude that are 3 clusters, but one has much more counts than the others. We can say that DBSCAN is not the best modeling algorithm to use since it's a density based algorithm and our data is very dense as we can see from the outputs, from std and from the other algorithms used.

Birch

Birch means Balanced Iterative Reducing and Clustering using Hierarchies, it's an algorithm that usually is used to cluster large datasets, it generates a small summary of the big dataset that retains as much information as possible. This smaller summary is then clustered instead of cluster the original and larger dataset.

Birch is often used as complement of other clustering algorithms, since birch can work efficiently with large amount of data sometimes it's used to get the summary of the dataset and then professional use other clustering techniques to cluster the data.

it's important to have in mind that Birch can only process metric attributes.

Birch has two important terms:

1. Clustering feature (CF): where birch summarizes the large data into smaller and dense regions. A CF entry can be composed by other entries.
2. CF Tree: Is a tree where each leaf node is considered a sub-cluster. Every CF tree contains a pointer to a child node and a CF entry made of the sum of CF entries in the child nodes.

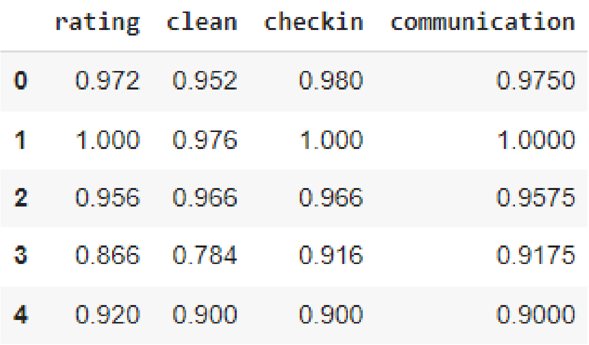
We can define the maximum number of entries in each leaf node by defining the threshold. Threshold, branching_factor and n_clusters are the parameters of birch.

Parameters:

- **Threshold:** Maximum number of data points a sub-cluster in a leaf node can hold.
- **branching_factor:** the maximum number of CF sub-clusters in each node
- **n_clusters:** The number of clusters to be returned after the entire BIRCH algorithm is complete

To execute the birch algorithm, we will use the dataset “LisbonDreams_preproF.csv”, since it's already preprocessed and scaled.

Since our project perspective is the Costumer Satisfaction Evaluation, first we selected the variables that are relevant for the analysis and we called it: df_birch



	rating	clean	checkin	communication
0	0.972	0.952	0.980	0.9750
1	1.000	0.976	1.000	1.0000
2	0.956	0.966	0.966	0.9575
3	0.866	0.784	0.916	0.9175
4	0.920	0.900	0.900	0.9000

After that, we imported the libraries that we require to run the birch. With everything set up, we can now create the Birch clustering model by defining the parameters:

1. **branching_factor = 50** - we kept the default value
2. **n_clusters = 3** - since the data is so dense and concentrated, we defined the number cluster on the output as 3.
3. **threshold = 0.05** - After trying different values like “0.1”, “0.2”, “0.5” and “1”, ”0.05” was the best output that we could get.

After this we fit and predicted the data and then created a visual output

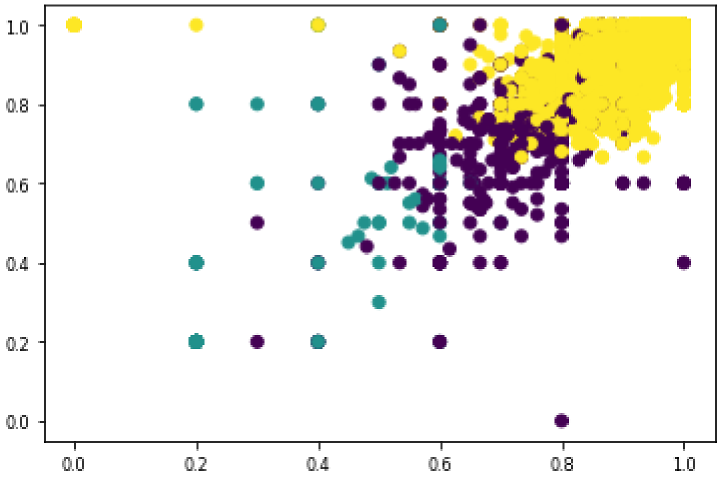


Figure 18 – Data Visual output

Looking at the final output, we can see that the data is concentrated and dense. We can see the three different clusters since it was the number that we attribute to the parameter “n_clusters”, considering the analysis done on hierarchical clustering and k-means.

We chose BIRCH after looking at the scikit-learn website comparing other algorithms applied to diverse types of data. Since Lisbon Dreams dataset is very dense, we saw birch as a good modeling option even though it's normally used for larger datasets.

Optics

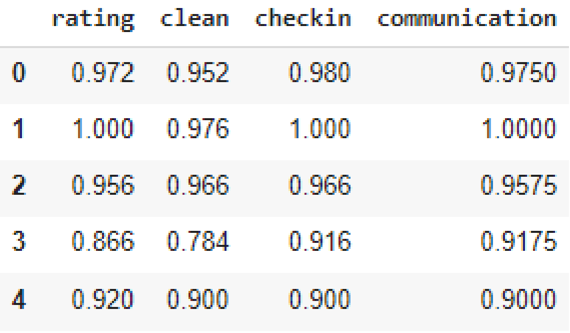
Optics stands for Ordering Points to Identify Cluster Structure. It has some inspiration on the DBSCAN algorithm, but it adds two more parameters:

- **Core Distance:** That is the minimum value of radius that a point requires to be considered as a core point.
- **Reachability Distance:** Expresses the distance which is reachable from a core point.

This technique is different from other clustering algorithms because this one does not segment data into clusters, but instead it produces a visualization of reachability distances and uses it to cluster de data.

To start we first imported all the needed libraries and selected the variables “rating”, “clean”, “checkin” and “communication” since those are the ones that are relevant for us taking into the account the project perspective.

Then we created a dataset called: df_opt



	rating	clean	checkin	communication
0	0.972	0.952	0.980	0.9750
1	1.000	0.976	1.000	1.0000
2	0.956	0.966	0.966	0.9575
3	0.866	0.784	0.916	0.9175
4	0.920	0.900	0.900	0.9000

Then we defined the parameters to build the clustering model:

- **min_samples = 50 -** we ketp it as 50 since it's the default value
- **xi = 0.1 -** it should be a float between 0 and 1. it defines the steepness on the reachability plot that constitutes a cluster boundary. Since our data is so dence we decided to choose 0.1 because it gaves better results.
- **min_cluster_size = 0.3** - it's the minimum number of samples in an OPTICS cluster and can be expressed as a faction of the number of samples. After trying different options 0.3 was the best value to use.

Since they are similar we decided to compare the outputs from the two different algorithms, OPTIC and DBSCAN.

We compared the outputs from OPTIC and DBSCAN with two different eps, 0.04 and 0.08.

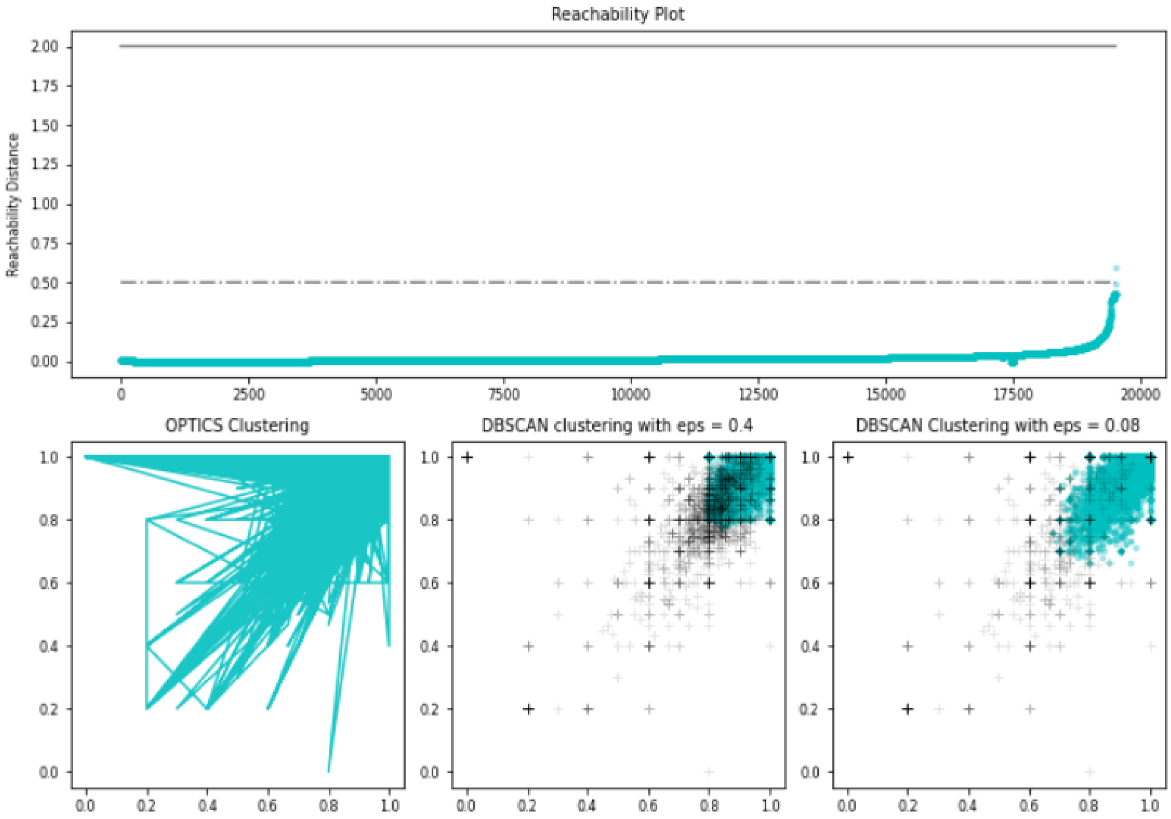


Figure 19 – OPTICS AND DBSCAN output comparison

After looking at the output we see the difference between the algorithms and on the DBSCAN we can see the cluster differences when we change the eps to double.

We can conclude say that the two algorithms are very similar because of the similar parameters that they share, like the epsilon and minimum number of points. The key difference is the that on DBSCAN we pick a fixed value.

Evaluation

Silhouette Score

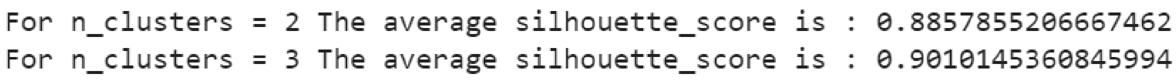
As part of the evaluation of the clusters that we found, we performed the silhouette score analysis, which is evaluates the goodness of the clustering algorithms through the calculation of the silhouette score:

$$s(i) = \frac{b(i) - a(i)}{\max\left(a(i), b(i)\right)}$$

It's important to state that these results are shown in values Until 1, which is the best value that a cluster can get. Then if the result is approaching 0, meaning that the clusters are overlapping and finally if we get negative values, it means that the sample has been assigned to the wrong cluster.

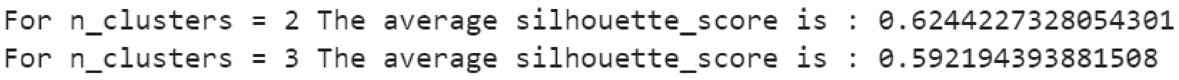
For this project we used the Silhouette score for 2 different algorithms, the ones that seem to work better for our data distribution, K-Means and SOM, also we tried the score for 2 clusters and 3 clusters because of the results in the elbow method. And the results are:

K-Means



```
For n_clusters = 2 The average silhouette_score is : 0.8857855206667462
For n_clusters = 3 The average silhouette_score is : 0.9010145360845994
```

SOM



```
For n_clusters = 2 The average silhouette_score is : 0.6244227328054301
For n_clusters = 3 The average silhouette_score is : 0.592194393881508
```

What we found in this calculation was that using 3 clusters to label the data was a good decision, because in the K-means algorithm obtaining a value of 0.9, incredibilly close to 1, that showed us the observations are impressive assigned to each of the clusters.

But in the case of the SOM algorithm the option with 3 cluster was not the one that has the highest value, but after comparing between the algorithms and understanding the data, we decided to keep the 3 clusters because, if we were to use 2 clusters it would be difficult to get insights from these 2 clusters, one of them would have too much observations and as the data is really dense, it's better to go in depth to analyze the small variations in the dataset, also the difference of 0.03 points between each number of clusters helped us to realize that our decision was correct.

Deployment

After applying different algorithms, one of the main conclusions is the data is very dense since the most important variables in our perspectives are having good qualifications, meaning that the observations have good ratings to describe the consumer's satisfaction. When we realized this about our dataset, we went in depth of the data to get more answers and being able to cluster our accommodations. In some algorithms we added more variables to our perspective, like the response and acceptance rate or even more. The result of the different modeling algorithms that we have run was that we should have 3 different clusters.

There is one cluster with most of the observations (88%) that has good ratings in general, but with opportunities to develop, for example the mean value of these observations in the cluster is the lowest, compared to the other clusters, but it's rating is always above 93/100. This cluster is the densest one and the standard deviation never gets higher then 0.1 (in a scale from 0 to 1).

The second densest cluster has 10% of the observations, this label has the best ratings in the 3 clusters, and has the lowest Standard deviations, which means that these accommodations have the best service and experience for the customer, this label covers the “star” accommodations in the portfolio of Dreams of Lisbon.

The last cluster is the one with fewer observations (3%) but it's still important because these accommodations are the ones that have a lot of potential, if we analyze the mean of the values of the ratings is not the lowest or the highest, but it has a lot of standard deviation, which means that working on the opportunities that we can find in the different kinds of variables, we can move these observations to the second cluster which has the best accommodations. It could be an effort but we could offer immediate better satisfaction to the customers if we work on this cluster.

Marketing Plan

The marketing plan must create an opportunity for the accommodations to move from the cluster of 88% and 3% of the observations to the one which has the best ratings (10% of the observations).

This would mean that we will be offering the best satisfaction experience for the customer.

To accomplish this, we decided on the following:

For the cluster with the most observations, we must work on the rating, because it's the lowest value of all the calcifications analyzed in our perspective. The main idea for improving this would be working in the advertisement of 25% of the accommodations, the ones that have the lowest mean of the qualifications. The way of working on this part of the cluster is by taking advantage of these observations having a good acceptance and response rate, so offering small discounts for the clients in their stays will help the owners to have more rotation of clients and with that experience we will learn from the needs of the new customers, also making these discounts would not be a loss, because we will be getting more volume of clients.

One of the most important points for the cluster, which is the shortest regarding the numbers of accommodations (3% of the accommodations), is to improve their response rate. For this case, it's stated that we already have potential clients, so, creating more advertisement would be recommended, but it will be better also to create a direct channel for the communication, so the owners and the company won't lose any customer that want to rent the accommodation.

For the last cluster, which is the one with 10% of the accommodations. The fact that the owners are not accepting some customers is having an impact on the rating qualification. it's important to state that this cluster has the accommodations with highest ratings, so a good marketing strategy for this cluster would be trying to impact clients that have good historical information about the places that they visited, meaning as these are our best accommodations, we must target these to the segment of customers that are really interested on the places.

Conclusion

We divide the different accommodations into 3 different clusters, based on the information that we found on the algorithms, these results show us opportunities within the clusters, and the main idea is to try to move the accommodations for the biggest and shortest cluster to the one that has the best rating, so the clients are able to feel an improvement in their customer journey while they choose “dreams of Lisbon” as their trust company to find accommodation. The strategy to make that happen would be the marketing campaign.